Rose II - an Automaton Based Evolution Approach

Daniel Doerr

March 30, 2010

1 Motivation

Since the first model of DNA evolution has been proposed by Jukes and Cantor in 1969 [2], a variety of more advanced models emerged, starting from the Kimura two-parameter model up to a 12-parameter model as well as models which also consider the non-independence of sites [1]. Similarly the models of protein evolution advanced, but unlike in DNA evolution models, site-dependency has been attached great importance due to the secondary and third structure of proteins.
Sequence generators for those models have been written and are extensively used (i.e. *seq-gen* [3]), more advanced sequence generators consider also insertions and deletions (such as in *Rose* [4]) as well as rearrangements and recombination, etc.
It is well known that different parts of genomic DNA evolve differently. Next to the distinction between non-coding areas, conserved regions such as protein coding and ribosomal genes, there are many other patterns which are conserved and evolve distinctively from others.
The implemented automaton based sequence generator allows to model DNA evolution in a very dynamic way such that evolution of different genomic regions can be modeled individually.

2 Results

For each edge in a given evolution tree, the automaton processes an ancestral sequence according to its annotations. That is, for each annotation, an automaton must be defined. At the time of writing, the automaton integration is still under construction, but it is planned to design automatons for all standard annotations, such that the user only has to specify annotations along a root sequence.
The states of the automaton are linked to mutations such as *insertion*, *deletion*, *substitution* or *match*, which absorbs its input symbols. It is planned to extend the list of operations to more advanced operations, which then also allow to simulate changes in substitution rate, nucleotide distribution, GTR violations etc.
Each automaton must have a *match* state, denoted **M**, from which other operations are accessible via a transition edge. The transitions are performed according to their probabilities. Optionally a start state can be defined, if a state other than the *match* state should be entered at the beginning of an annotation.
The retention time in each state is drawn according to a distribution function which can vary in each state, except for **M**. Here, the retention time is modeled according to the evolution tree, such that on average the number of mutations equals the edge length from the ancestor to the currently processed descendant. That is, as many *"runs"* from the match state through mutation states are performed as the length of corresponding edge.
When a sequence with several annotations is given, the *Automaton Controller* processes the sequence and relays each annotated region to its corresponding automaton with adapted mutation rates, such that on average the edge length equals the sum of mutations in all annotations.

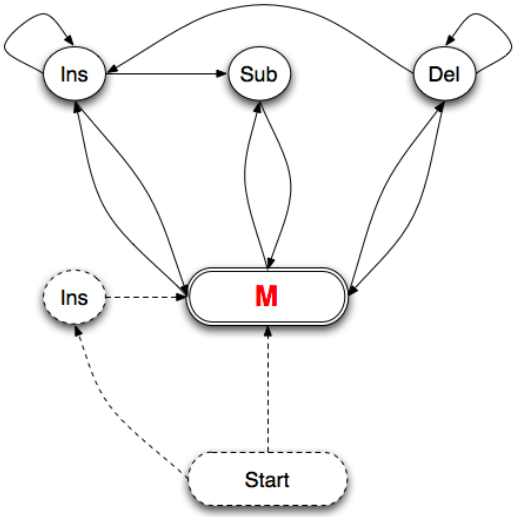

Figure 1: Example automaton. *Ins, Del, Sub* and *M* denote operations *insertion, deletion, substitution* and *match*, respectively. The optional definition of a *start* state is also indicated by the states and edges in dashed lines.

3 Outlook

Automaton based sequence evolution establishes new opportunities to explore models of evolution which behave according to well defined but unconventional patterns that may violate phylogenetic assumptions of the models mentioned above.
The current implementation already allows to design powerful automatons. It is up to future projects to model meaningful automatons which correspond to the current set of default annotations and beyond.

References

[1] J. Felsenstein. *Inferring Phylogenies*. Sinauer Associates, 2 edition, September 2003.

[2] T. H. Jukes and C. R. Cantor. *Evolution of Protein Molecules*. Academy Press, 1969.

[3] A. Rambaut and N. C. Grassly. Seq-gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Comput Appl Biosci*, 13(3):235–238, Jun 1997.

[4] J. Stoye, D. Evers, and F. Meyer. Rose: generating sequence families. *Bioinformatics*, 14(2):157–163, 1998.